Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Identification of a Class of Discrete Event Systems by Neural Networks - Sparse Realization -

Yasuaki Kuroe *, **Yoshihiro Mori** *

* *Kyoto Institute of Technology, Kyoto, 606-8585 Japan (Tel: +81-75-724-{7445,7463}; e-mail: {kuroe,yosihiro}@kit.ac.jp).*

Abstract: In analysis and design of a system, the first goal is to obtain an appropriate model of the system. Because of the complex dynamics of discrete event systems (DESs), it is very difficult to obtain a model of unknown DESs from given input and output data. This paper discusses an identification and realization method of a class of DESs by neural networks. We consider a class of DESs which is modeled by using finite state automata. In identification and realization of systems by using neural networks, it is essentially important to develop a suitable architecture of neural networks. We already proposed two neural network architectures: one is a class of recurrent neural networks and the other is a class of recurrent high-order neural networks, which are capable of representing FSA with the network size being smaller than the existing neural network models. In this paper we present an identification method of DESs, which makes it possible to obtain sparse realization, that is, to obtain networks with simpler structure. It is shown through numerical experiments that presented method makes it possible to obtain simpler neural networks which can exactly simulate target DESs.

1. INTRODUCTION

Discrete event systems (DESs) are dynamic systems with state changes driven by occurrence of events and many practical systems can be modeled as DESs. Recently, there have been increasing research interests of DESs and also hybrid systems which are those that combine continuous and discrete dynamics and involve both variables that take values in a continuum and those that take values in a finite or countable set. In many control systems, controlled objects are continuous dynamical systems and controllers are implemented as discrete event systems.

In analysis and design of any system, the first goal is to obtain an appropriate mathematical model of the system. Because of the complex dynamics of DESs, it is very difficult to obtain a mathematical model of unknown DESs. Therefore it has been strongly desired to develop identification and realization methods of DESs from given input and output data. One of the promising approaches to the problem is to develop a method by using neural networks. In this paper we discuss an identification and realization problem of a class of DESs by neural networks. We consider a class of DESs represented by finite state automata (FSA), which is one of the most familiar model representations of DESs.

In recent years, the problem of representing and learning FSA with artificial neural networks has attracted a great deal of interest in the area of brain science. Several models of neural networks for representing and learning FSA have been proposed and their computational capabilities have been investigated (Minsky [1967], Alon et al. [1991], Giles et al. [1992], Zegn et al. [1993], Giles et al. [1995], Kuroe [2005]).

The first problem that comes out in the approach is to investigate what architectures of neural networks are suitable for identification and realization of DESs. We have already proposed architectures of neural networks which are suitable for identification and realization of a class of DESs represented by FSA (Kuroe et al. [2007]). Two neural network models are proposed: one is a class of recurrent neural networks (RNNs) and the other is a class of recurrent high-order neural networks (RHONNs). They are both capable of representing the DESs with the network size being smaller than the existing models.

In this paper we present an identification method of the DESs from a given set of input and output data by using those neural networks. In those neural network models there are a lot of zero-valued parameters which are to be determined in identification. It is, therefore, essentially important to obtain sparse realization, that is, to obtain a neural network with as many zero-valued parameters as possible. The identification problem is reduced to a learning problem of the neural networks. The presented method makes it possible to obtain sparse realization by introducing a cost function which reflects a measure of network complexity in formulation of the learning problem. We perform several identification experiments to verify performance of the presented method. It is shown through the experiments that the proposed method successfully obtains sparse realization of the target DESs.

2. PROBLEM FORMULATION

2.1 Model of Discrete Event Systems

In this paper we consider a class of DESs $\mathcal{M}$ described by finite state automata, which is defined by

$$\mathcal{M} = (Q, q_0, \Sigma, \Delta, \delta, \varphi) \tag{1}$$

where Q is the set of state symbols: $Q = \{q^{(1)}, q^{(2)}, \cdots, q^{(r)}\}$, r is the number of state symbols, $q_0 \in Q$ is the initial state, Σ is the set of input symbols: $\Sigma = \{i^{(1)}, i^{(2)}, \cdots, i^{(m)}\}$, m is the number of input symbols, Δ is the set of output symbols: $\Delta = \{o^{(1)}, o^{(2)}, \cdots, o^{(l)}\}$, l is the number of output symbols, δ: $Q \times \Sigma \to Q$ is the sate transition function and φ: $Q \times \Sigma \to \Delta$ is the output function.

We suppose that a DES $\mathcal{M}$ operates at unit time intervals. Letting $i(t) \in \Sigma$, $o(t) \in \Delta$ and $q(t) \in Q$ be the input symbol, output symbol and state symbol at time t, respectively, then a DES $\mathcal{M}$ is described by the discrete dynamical system of the form:

$$\mathcal{M} : \begin{cases} q(t+1) = \delta(q(t), i(t)), \ q(0) = q_0 \\ o(t) = \varphi(q(t), i(t)) \end{cases} \tag{2}$$

The objective of this paper is to discuss the identification and realization problem of the DESs described by (1) or (2) by using neural networks.

2.2 Identification and Realization of Discrete Event Systems

We now formulate an identification and realization problem of unknown DES described by (1) or (2). We assume the followings.

Assumption 1. The set of state symbols Q and the initial state q_0 of the DES are unknown.

Assumption 2. The state transition function δ and output function φ of the DES are unknown.

Assumption 3. The sets of input symbols Σ and output symbols Δ of the DES are known.

Assumption 4. A set of data of input sequences $\{i(t)\}$ and the corresponding output sequences $\{o(t)\}$ are available.

Those assumptions are natural in the real identification and realization problems of DESs. The problem is formulated as follows. Given a set of data of input and output sequences, $\{i(t)\}$ and $\{o(t)\}$, of a target DES $\mathcal{M}$, determine structures and values of the parameters of a neural network such that its input and output relation becomes equal to that of the DES. Two problems arise in the identification and realization problem of the DESs: the first one is how to determine architectures of neural networks suitable for identification and realization, and the second one is how to determine values of their parameters, that is, those of connection weights and threshold values. We first discuss how to determine suitable architectures of neural networks.

3. RECURRENT NEURAL NETWORKS WITH HIGH-ORDER CONNECTIONS

We introduce a general class of neural networks in order to derive suitable architectures of neural networks for identification and realization of the DESs $\mathcal{M}$ described by (1) or (2). The neural networks considered here possess not only usual connection units but also high-order connection units (Kosmatopoulos et al. [1995], Kuroe et al. [1997]). Figure 1 shows the schematic diagram of the neural network model considered in this paper. The network consists of neuron units, connection units, external inputs and external outputs. Two types of neurons are considered: dynamic neurons and static neurons. All the neurons and connection units are arbitrarily connected, and the connection units allow high-order nonlinear interactions.

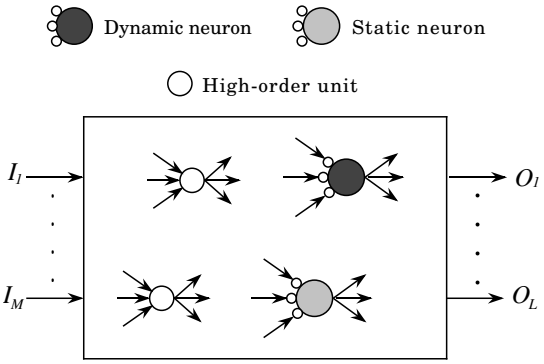

Fig. 1. Recurrent neural networks with high-order connection units.

Let N_d, N_s, N_c, M and L be the numbers of the dynamic neurons, the static neurons, the connection units, the external inputs and the external outputs existing in the network, respectively. The mathematical model of the dynamic neurons is given by

$$v_i^d(t+1) = \sum_{j=1}^{N_d} w_{ij}^{dd} h_j^d(t) + \sum_{j=1}^{N_s} w_{ij}^{ds} h_j^s(t) + \sum_{j=1}^{N_c} w_{ij}^{d} z_j(t) + \sum_{j=1}^{M} w_{ij}^{dI} I_j(t) + \theta_i^d, \quad v_i^d(0) = v_{i0}^d \tag{3}$$

$$h_i^d(t) = f_i^d(v_i^d(t)), \qquad (i = 1, 2, \cdots, N_d) \tag{4}$$

and the mathematical model of the static neurons is given by

$$u_i^s(t) = \sum_{j=1}^{N_s} w_{ij}^{ss} h_j^s(t) + \sum_{j=1}^{N_d} w_{ij}^{sd} h_j^d(t) + \sum_{j=1}^{N_c} w_{ij}^{s} z_j(t) + \sum_{j=1}^{M} w_{ij}^{sI} I_j(t) + \theta_i^s \tag{5}$$

$$h_i^s(t) = f_i^s(u_i^s(t)), \qquad (i = 1, 2, \cdots, N_s) \tag{6}$$

where $v_i^d(t)$, v_{i0}^d, $h_i^d(t)$ and θ_i^d are the state, the initial state, the output and the threshold value of the i-th dynamic neuron, respectively, and u_i^s, h_i^s and θ_i^s are the state, the output and the threshold value of the i-th static neuron, respectively. $z_j(t)$ is the output of the j-th connection unit, I_j is the j-th external input, w_{ij}^{dd} is the weight from the j-th dynamic neuron to the i-th dynamic neuron, w_{ij}^{ds} is the weight from the j-th static neuron to the i-th dynamic neuron, w_{ij}^{d} is the weight from the j-th connection unit to the i-th dynamic neuron, w_{ij}^{dI} is the weight from the j-th external input to the i-th dynamic neuron, w_{ij}^{ss} is the weight from the j-th static neuron to the i-th static neuron, w_{ij}^{sd} is the weight from the j-th dynamic neuron to the i-th static neuron, w_{ij}^{s} is the weight from the j-th connection unit to the i-th static neuron, w_{ij}^{sI} is the weight from the j-th external input to the i-th static neuron. $f_i^d(\cdot)$

and $f_i^s(\cdot)$ are nonlinear output functions of the dynamic and static neurons such as threshold or sigmoidal function.

The connection units determine connections among the neurons and the external inputs, in which high-order nonlinear interactions are allowed. Let $\boldsymbol{g}$ be a N_d+N_s+M dimensional vector defined by

$$\boldsymbol{g} = [h_1^d, h_2^d, \cdots, h_{N_d}^d, h_1^s, h_2^s, \cdots, h_{N_s}^s, I_1, I_2, \cdots, I_M]^T. \quad (7)$$

The model of the high-order connection units is given by the following equation.

$$z_j(t) = G_j(\boldsymbol{g}(t)), \qquad (j = 1, 2, \cdots, N_c) \quad (8)$$

where $G_j(\boldsymbol{g}(t))$ is defined by

$$G_j(\boldsymbol{g}(t)) := \prod_{k \in J_j} g_k^{d_k(j)}(t).$$

In the above equation $g_k(t)$ is the k-th element of $\boldsymbol{g}(t)$, J_j is a subset of the index set $\{1, 2, \cdots, N_d + N_s + M\}$ and $d_k(j)$ is a nonnegative integer. Note that $d_k(j)$ is an exponent, not a superscript.

The external outputs O_i are expressed by

$$O_i(t) = \sum_{j=1}^{N_d} \delta_{ij}^d h_j^d(t) + \sum_{j=1}^{N_s} \delta_{ij}^s h_j^s(t) \quad (i = 1, 2, \cdots, L) \quad (9)$$

where δ_{ij}^d and δ_{ij}^s take values 1 or 0. If the output of the j-th dynamic (static) neuron is connected to the i-th external output, $\delta_{ij}^d = 1$ ($\delta_{ij}^s = 1$), otherwise $\delta_{ij}^d = 0$ ($\delta_{ij}^s = 0$).

4. MODELS OF NEURAL NETWORKS FOR IDENTIFICATION OF DISCRETE EVENT SYSTEMS

There have been done several works on the representation of FSA with neural networks or on investigation of relationship between neural network architectures and FSA. A typical representative of neural network architectures for representing FSA is a class of second-order neural networks (Giles et al. [1992, 1995]). In these neural network models, each symbol of input, state and output symbols is expressed by unit basis vector, for example, each state symbol $q^{(i)}$ be expressed by r dimensional unit basis vector, that is $q^{(1)} = (1, 0, \cdots, 0)$, $q^{(2)} = (0, 1, \cdots, 0)$, $\cdots$, $q^{(r)} = (0, 0, \cdots, 1)$ and state of FSA is represented by assigning one neuron individually. Then, as the number of states of a target FSA increases, the number of neurons required for representing the FSA increases, which makes it difficult to identify the FSA because of a large number of network parameters.

We have already proposed two architectures of neural networks for representing a DES $\mathcal{M}$ described by FSA, which resolve this problem: one is a class of recurrent neural networks without high-order connections and the other is a class of recurrent neural networks with high-order connections (Kuroe et al. [2007]).

4.1 Recurrent Neural Networks

We first show a model of recurrent neural networks (RNN) without high-order connections for identification and realization of a DES $\mathcal{M}$. We encode all the state symbols $q^{(i)}$ $(i = 1, 2, \cdots, r)$, input symbols $i^{(i)}$ $(i = 1, 2, \cdots, m)$ and output symbols $o^{(i)}$ $(i = 1, 2, \cdots, \ell)$ of the DES as binary variables. For example, if the number of state symbols of a DES is four ($r = 4$), $q^{(1)} = (0, 0)$, $q^{(2)} = (0, 1)$, $q^{(3)} = (1, 0)$ and $q^{(4)} = (1, 1)$. Then $q(t)$, $i(t)$ and $o(t)$ in (2) can be expressed as follows.

$$\begin{aligned} q(t) &= (s_1(t), s_2(t), \cdots, s_\alpha(t)) \quad (s_i(t) \in \{0, 1\}) \\ i(t) &= (x_1(t), x_2(t), \cdots, x_\beta(t)) \quad (x_i(t) \in \{0, 1\}) \\ o(t) &= (y_1(t), y_2(t), \cdots, y_\gamma(t)) \quad (y_i(t) \in \{0, 1\}) \end{aligned} \quad (10)$$

where α, β and γ are natural numbers, which are determined depending on r, m and l, respectively, that is, α is the minimum natural number satisfying $r \leq 2^\alpha$, β is the minimum natural number satisfying $m \leq 2^\beta$ and γ is the minimum natural number satisfying $l \leq 2^\gamma$. Expressing all the input, state and output symbols of the DES by binary variables as shown in (10), its state transition function and output function are expressed as Boolean functions. Let z_i, $i = 1, 2, \cdots, n$ $(n := \alpha + \beta)$ be defined by $z_1 = s_1, z_2 = s_2, \cdots, z_\alpha = s_\alpha, z_{\alpha+1} = x_1, z_{\alpha+2} = x_2, \cdots, z_n = x_\beta$. By using the fact that (i) any Boolean function can be represented in conjunctive normal form and (ii) letting 'true = 1' and 'false = -1', basic logical operations AND, OR and NOT can be expressed by combining the arithmetic operations and the sign function $S(\cdot)$ defined by $S(x) = 1$ for $x \geq 0$ and $S(x) = -1$ for $x < 0$, we can transform the model of the DES (eq.(2)) into the following equation.

$$\mathcal{M}: \begin{cases} s_i(t+1) = S(\sum_{j=1}^{2^n} a_{ij} Z_j(t) + n_i^s - 1) \\ \qquad\qquad (i = 1, 2, \cdots, \alpha) \\ y_i(t) = S(\sum_{j=1}^{2^n} b_{ij} Z_j(t) + n_i^y - 1) \\ \qquad\qquad (i = 1, 2, \cdots, \gamma) \end{cases} \quad (11)$$

where

$$\begin{aligned} Z_1(t) &= S(z_1(t) + \cdots + z_{n-1}(t) + z_n(t) - (n-1)) \\ Z_2(t) &= S(z_1(t) + \cdots - z_{n-1}(t) + z_n(t) - (n-1)) \\ &\vdots \\ Z_{2^n}(t) &= S(-z_1(t) - \cdots - z_{n-1}(t) - z_n(t) - (n-1)) \end{aligned} \quad (12)$$

and n_i^s and n_i^y are the number of the elements of $\{a_{ij} : a_{ij} = 1, i = 1, 2, \cdots, \alpha, j = 1, 2, \cdots, 2^n\}$ and $\{b_{ij} : b_{ij} = 1, i = 1, 2, \cdots, \gamma, j = 1, 2, \cdots, 2^n\}$, respectively. Based on (11) we can derive an architecture of neural networks which can exactly represent the DES as follows.

Consider the neural network described by (3), (4), (5), (6), (7), (8) and (9) where we let $N_d = \alpha$, $N_s = 2^{\alpha+\beta} + \gamma$, $N_c = 0$, $M = \beta$, $L = \gamma$ and $f_i^d(\cdot) = S(\cdot)$ and $f_i^s(\cdot) = S(\cdot)$. In the networks we let the state vector of the dynamic neurons $\boldsymbol{v}^d = (v_1^d, v_2^d \cdots, v_\alpha^d)$, the external input vector $\boldsymbol{I} = (I_1, I_2, \cdots, I_\beta)$, and the external output vector $\boldsymbol{O} = (O_1, O_2, \cdots, O_\gamma)$ be corresponding to the state q, the input i and the output o of the DES where they are encoded as binary variables. To realize the first equation of (11), α dynamic neurons described by (3) and (4) are assigned, in which we let $w_{ij}^{dd} = 0$ for $i, j = 1, 2 \cdots, \alpha$, $w_{ij}^{ds} = 0$ for $i = 1, 2 \cdots, \alpha$, $j = 2^{\alpha+\beta} + 1, 2^{\alpha+\beta} + 2, \cdots, 2^{\alpha+\beta} + \gamma$, and

$w_{ij}^{dI} = 0$ for $i = 1, 2\cdots, \alpha,\ j = 1, 2, \cdots, \beta$. Note that the values of θ_i^d $(i = 1, 2, \cdots, \alpha)$ can be determined uniquely from the values of w_{ij}^{ds} $(i = 1, 2\cdots, \alpha,\ j = 1, 2, \cdots, 2^{\alpha+\beta})$ by the definition of n_i^s in (11). To realize (12), $2^{\alpha+\beta}$ static neurons described by (5) and (6) are assigned, in which we let $w_{ij}^{ss} = 0$ for $i, j = 1, 2\cdots, 2^{\alpha+\beta}$. Note that the values of w_{ij}^{sd} $(i = 1, 2\cdots, 2^{\alpha+\beta},\ j = 1, 2\cdots, \alpha)$, w_{ij}^{sI} $(i = 1, 2\cdots, 2^{\alpha+\beta},\ j = 1, 2\cdots, \beta)$ and θ_i^s $(i = 1, 2\cdots, 2^{\alpha+\beta})$ can be determined from (12); w_{ij}^{sd} and w_{ij}^{sI} take the values of '1' or '0' and $\theta_i^s = -(n-1)$ $(i = 1, 2\cdots, 2^{\alpha+\beta})$. To realize the second equation of (11), additional γ static neurons described by (5) and (6) are assigned, in which we let $w_{ij}^{ss} = 0$ for $i = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma,\ j = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma$, $w_{ij}^{sd} = 0$ for $i = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma,\ j = 1, 2, \cdots, \alpha$ and $w_{ij}^{sI} = 0$ for $i = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma,\ j = 1, 2, \cdots, \beta$. Note that the values of θ_i^s $(i = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma)$ can be determined uniquely from the values of w_{ij}^{ss} $(i = 2^{\alpha+\beta}+1, 2^{\alpha+\beta}+2, \cdots, 2^{\alpha+\beta}+\gamma,\ j = 1, 2, \cdots, 2^{\alpha+\beta})$ by the definition of n_i^y in (11). Furthermore the external outputs described by (9) are assigned, in which we let $\delta_{ij}^d = 0$ for $i = 1, 2\cdots, \gamma,\ j = 1, 2, \cdots, \alpha$ and $\delta_{ij}^s = 0$ for $i = 1, 2\cdots, \gamma,\ j = 1, 2, \cdots, 2^{\alpha+\beta}$. Figure 2 shows the recurrent neural network thus constructed, which consists of α dynamic neurons, and $2^{\alpha+\beta}+\gamma$ static neurons. It can be shown by using (11) and (12) that the neural networks thus constructed, the recurrent neural network (RNN), are capable of strictly representing the DES.

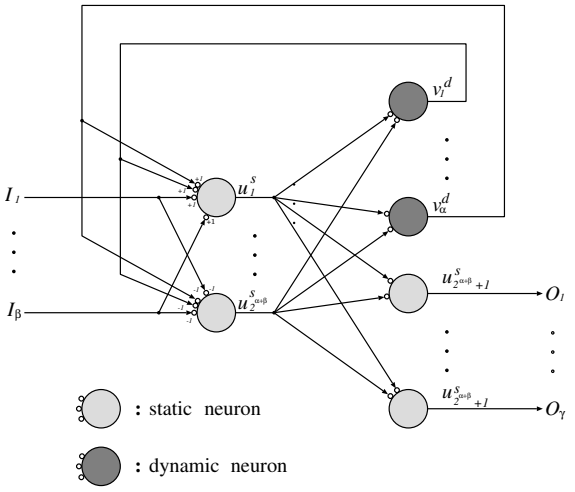

Fig. 2. Recurrent neural network for representing the DES

4.2 Recurrent High-Order Neural Networks

Here we show a model of recurrent neural networks with high-order connections, a recurrent high-order neural network (RHONN) for identification and realization of a DES $\mathcal{M}$. We encode all the state symbols $q^{(i)}$ $(i = 1, 2, \cdots, r)$, input symbols $i^{(i)}$ $(i = 1, 2, \cdots, m)$ and output symbols $o^{(i)}$ $(i = 1, 2, \cdots, \ell)$ of the DES as binary variables shown in (10). Similar to the above subsection, by using the fact that (i) any Boolean function can be represented in conjunctive normal form and (ii) basic logical operations AND, OR and NOT can be expressed by combining the arithmetic operations and the Heviside function $H(\cdot)$ defined by $H(x) = 1$ for $x \geq 0$ and $H(x) = 0$ for $x < 0$, we can transform the model of the DES (eq.(2)) into the following equation.

$$\mathcal{M}: \begin{cases} s_i(t+1) = H(\sum_{j=1}^{2^n-1} a_{ij}^* z_j^*(t) + a_{i2^n} - \gamma_i^q), \\ \qquad s_i(0) = s_{i0} \quad (i = 1, 2, \cdots, \alpha) \\ y_i(t) = H(\sum_{j=1}^{2^n-1} b_{ij}^* z_j^*(t) + b_{i2^n} - \gamma_i^y) \\ \qquad (i = 1, 2, \cdots, \gamma) \end{cases} \tag{13}$$

where

$$\begin{cases} z_1^*(t) = z_1(t)z_2(t)z_3(t)\cdots z_n(t) \\ z_2^*(t) = z_2(t)z_3(t)\cdots z_n(t) \\ z_3^*(t) = z_1(t)z_3(t)\cdots z_n(t) \\ \quad \vdots \\ z_{2^n-1}^*(t) = z_n, \end{cases} \tag{14}$$

a_{ij}^* and b_{ij}^* are integers, and γ_i^q and γ_i^y are real numbers satisfying $0 < \gamma_i^q \leq 1$ and $0 < \gamma_i^y \leq 1$. Based on (13) we can derive an architecture of neural networks which can exactly represent the DESs as follows.

Consider the neural network described by (3), (4), (5), (6), (7), (8) and (9) where we let $N_d = \alpha$, $N_s = \gamma$, $N_c = 2^n - 1$ $(n = \alpha + \beta)$, $M = \beta$, $L = \gamma$ and $f_i^d(\cdot) = H(\cdot)$ and $f_i^s(\cdot) = H(\cdot)$, and $\boldsymbol{g}$ in (7) is defined by

$$\boldsymbol{g} = [h_1^d, h_2^d, \cdots, h_\alpha^d, I_1, I_2, \cdots, I_\beta]^T. \tag{15}$$

In the dynamic neurons we let $w_{ij}^{dd} = 0$ for $i, j = 1, 2, \cdots, \alpha$, $w_{ij}^{ds} = 0$ for $i = 1, 2, \cdots, \alpha, j = 1, 2, \cdots, \gamma$ and $w_{ij}^{dI} = 0$ for $i = 1, 2, \cdots, \alpha, j = 1, 2, \cdots, \beta$, and in the static neurons we let $w_{ij}^{ss} = 0$ for $i, j = 1, 2, \cdots, \gamma$, $w_{ij}^{sd} = 0$ for $i = 1, 2, \cdots, \gamma, j = 1, 2, \cdots, \alpha$ and $w_{ij}^{sI} = 0$ for $i = 1, 2, \cdots, \gamma, j = 1, 2, \cdots, \beta$. Furthermore, in the high order connection units (8), for each z_j^* defined by (14), we define an index set J_j^* consisting of indexes of the elements of the monomial, that is, $J_1^* = \{1, 2, 3, \cdots, n\}$, $J_2^* = \{2, 3, \cdots, n\}$, $\cdots$, $J_{2^n-1}^* = \{n\}$ and we let $J_j = J_j^*$ for $j = 1, 2, \cdots, 2^n - 1$ and $d_k(j) = 1$ for $j = 1, 2, \cdots 2^n - 1$. In the external outputs (9), δ_{ij}^d and δ_{ij}^s are chosen as $\delta_{ij}^d = 0$ for $i = 1, 2, \cdots, \gamma, j = 1, 2, \cdots, \alpha$ and $\delta_{ij}^s = 1$, for $j = i$, $\delta_{ij}^s = 0$ for $j \neq i, i = 1, 2, \cdots, \gamma$. Figure 3 shows the recurrent high-order neural network thus constructed, which contain at most n-th order product connections in their connection units. It can be shown that the RHONN can represent the DES, by using the expressions (13) and (14) and by letting the states $v_i^d(t)$ of the dynamic neurons, the external inputs I_i and the external outputs O_i be corresponding to the states $q_i(t)$, the inputs $i_i(t)$ and the outputs $o_i(t)$ of the DES.

5. IDENTIFICATION OF DISCRETE EVENT SYSTEMS

5.1 Method

In this section we discuss the identification and realization problem of unknown DES by using the RNNs and

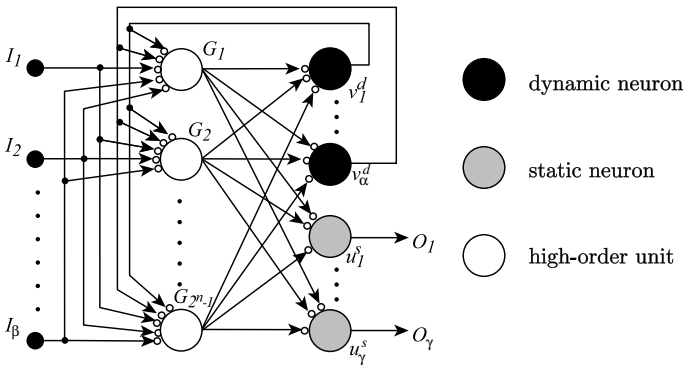

Fig. 3. Recurrent high-order neural network for representing the DES

RHONNs presented in the above section. For identification and realization we construct a RNN or RHONN in the manner discussed in the previous section. Note that, β and γ can be uniquely determined since the number of the input states and the output states of a target DES are known, on the other hand, α cannot be determined since the number of the states of the DES is not known. That is to say, in the RNN the number of the external inputs and outputs can be uniquely determined as $M = \beta$ and $L = \gamma$ a priori, on the other hand, the number of the dynamic neurons ($N_d = \alpha$) and the static neurons ($N_s = 2^{\alpha+\beta} + \gamma$) cannot be determined. In the RHONN, the number of the static neurons, the number of external inputs and the number of the external outputs can be determined uniquely as $N_s = \gamma$, $M = \beta$ and $L = \gamma$, but the number of dynamic neurons $N_d = \alpha$ and the number of the high-order connection units $N_c = 2^n - 1, (n = \alpha + \beta)$ cannot be uniquely determined. We provide the number of dynamic neurons (so the number of the static neurons in the RNN and that of the high-order units in the RHONN) large enough so that N_d becomes greater than (or hopefully equal to) α of the target DES.

By using the neural network thus constructed for a target DES, the identification problem is solved in the following manner. For a set of data of input sequences $\{i(t) : t = 0, 1, 2, \cdots, t_f\}$ and the corresponding output sequences $\{o(t) : t = 0, 1, 2, \cdots, t_f\}$ of the target DES where t_f is the length of the sequences, we give the input $i(t)$ of the target DES which is encoded as a binary variable to the input of the neural network $I_j(t)$, $I_j(t) = i_j(t), j = 1, 2, \cdots, M(= \beta)$ and train the neural network, in such a way that the corresponding output of the neural network $O_i(t)$ becomes equal to the output data $o_i(t)$, $O_i(t) = o_i(t)$. The problem is to determine values of the parameters of the neural network so as to minimize:

$$\frac{1}{2}\sum_{t=0}^{t_f}\sum_{i=1}^{L}\left|o_i(t) - O_i(t)\right|^2.$$

Note that in the neural network models presented in the previous section there are a lot of zero-valued parameters which are to be determined. It is, therefore, essentially important to obtain sparse realization, that is, to obtain a network with as many zero-valued parameters as possible. For this purpose we introduce an additional cost function wihch reflects complexity of the neural network.

Define the performance index by

$$J = J_1 + \rho J_2 \tag{16}$$
$$J_1 = \frac{1}{2}\sum_{t=0}^{t_f}\sum_{i=1}^{L}\left|o_i(t) - O_i(t)\right|^2$$
$$J_2 = \|\boldsymbol{w}\|_1 = \sum_{k,\ell=d,s}\sum_{i,j}\left|w_{ij}^{k\ell}\right| + \sum_{k=d,s}\sum_{i,j}\left|w_{ij}^{k}\right|$$

where $\rho > 0$ is a weighting coefficient. Note that J_2 is the ℓ^1 norm of the parameter vector of the neural network, which can be a measure of complexity of the network. In Ishikawa [1996], it is shown that, in the learning problem of neural networks, the choice of ℓ^1 norm of the parameter vector as the cost function brings much fewer nonzero parameters than that of Euclidean norm $\|\boldsymbol{w}\|_2$. Therefore the cost function J_2 could bring a sparse realization of the network.

The problem now is reduced to a learning problem of neural networks, that is to finding values of the parameters of the RNN or RHONN which minimize the performance index J. Note that the nonlinear functions $S(\cdot)$ and $H(\cdot)$ are not differentiable, which implies that the gradient-based algorithms such as the steepest descent method cannot be applied to the optimization problem. We replace these nonlinear functions of each neuron by smooth sigmoidal functions which can approximate them with reasonable accuracy and utilize the gradient-based algorithms in which several useful algorithms are available: the steepest descent algorithm, the conjugate gradient algorithm, the quasi-Newton algorithm and so on. For instance, the steepest descent algorithm is described as follows.

$$\boldsymbol{\omega}^{k+1} = \boldsymbol{\omega}^k - \eta \cdot \frac{\partial J}{\partial \boldsymbol{\omega}^k} \tag{17}$$

where $\boldsymbol{\omega}$ is the parameter vector. The main problem associated with these algorithms is the computation of the gradients $\partial J/\partial \boldsymbol{\omega}$. The gradients can be calculated by deriving adjoint neural networks of the RHONNs (Kuroe et al. [1997]). The derivation of the gradients is omitted here.

Note also that the initial values of the dynamic neurons $v_i^d(0)$ can not be given a priori because of the assumption that the initial states of the DES are unknown. Therefore we choose as the learning parameters not only the connection weights and the threshold values but also the initial states of the dynamic neurons. In the RNN, the learning parameters are w_{ij}^{ds} $(i = 1, 2\cdots, \alpha,\ j = 1, 2, \cdots, 2^{\alpha+\beta})$ and w_{ij}^{ss} $(i = 2^{\alpha+\beta} + 1, 2^{\alpha+\beta} + 2, \cdots, 2^{\alpha+\beta} + \gamma,\ j = 1, 2, \cdots, 2^{\alpha+\beta})$ and values of the initial conditions of the dynamic neurons v_{i0}^d $(i = 1, 2\cdots, \alpha)$, the total number of which is $2^{(\alpha+\beta)} \times (\alpha+\gamma) + \alpha$. In the RHONN, the learning parameters are w_{ij}^d, θ_i^d, w_{ij}^s, θ_i^s and $v_i^d(0)$, the total number of which is also $2^{(\alpha+\beta)} \times (\alpha + \gamma) + \alpha$.

5.2 Numerical Experiments

Here we present experimental results of identification and realization of the DESs $\mathcal{M}$ by using the RNNs or RHONNs. We have performed identification for three examples of the DESs. The first one is a simple DES whose state transition diagram is shown in Fig. 4. This DES accepts the sequences consisting of only '1'. The number of the state symbols of the DES is two and $\Sigma = \Delta = \{0, 1\}$.

The second example is a DES whose state transition diagram is shown in Fig. 5, which accepts the sequence $(10)^*$. The number of the state symbols of the DES is three and $\Sigma = \Delta = \{0, 1\}$. The third one is a DES whose state transition diagram is shown in Fig. 6. This DES accepts the sequences which do not include '000'. The number of state symbols of the DES is four and $\Sigma = \Delta = \{0, 1\}$.

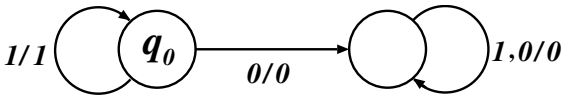

Fig. 4. Example 1: DES accepting the sequences consisting of only '1'

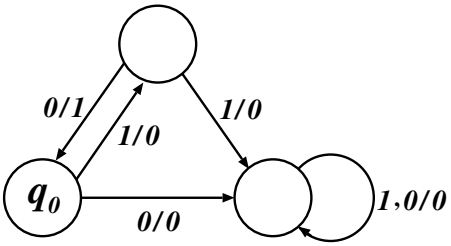

Fig. 5. Example 2: DES accepting the sequence $(10)^*$.

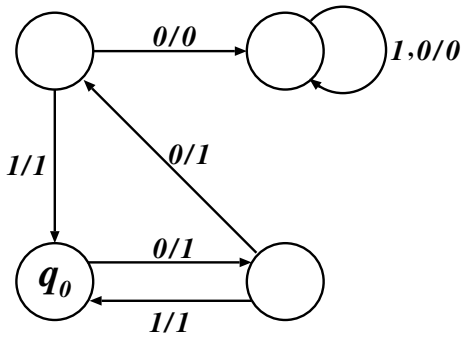

Fig. 6. Example 3: DES accepting the sequences that do not include '000'

For identification of these DESs, we construct the RNNs and RHONNs in the manner discussed in §4.1 and §4.2. Noting the assumption that the set of the state symbols and the initial state of the DESs are not known, only the input and output data are available, we can determine uniquely the number of the static neurons, the number of external inputs $M = \beta$ and the number of the external outputs $L = \gamma$ in the RNNs and RHONNs, but we cannot determine the number of dynamic neurons $N_d = \alpha$. In these examples we can choose $M = \beta = 1$ and $L = \gamma = 1$. We have performed experiments with changing the number of dynamic neurons $N_d = \alpha$. In the learning of the RNNs and RHONNs we use the smooth nonlinear function $\frac{2}{1-\exp(-ax)} - 1$ for $S(\cdot)$ and $\frac{1}{1-\exp(-ax)}$ for $H(\cdot)$, where a is a parameter. It is known that FSA with r state symbols is uniquely identified by using all input sequences of length $2r-1$. We train the neural networks by using all sequences of length $2r-1$ as learning data. We use the conjugate gradient method to minimize the performance index (16).

Tables 1 and 2 show the minimum values of the performance indexes J, J_1 and J_2 obtained by the learning experiments of the RNNs and RHONNs where we let $a = 1$ and $\rho = 0.01$. It has been checked that in all the experiments the target DES are successfully identified and the obtained neural networks can perform exactly the same operations as the target DESs do by considering some tolerances to the threshold values (1 and -1 for RNNs and 1 and 0 for RHONNs), which are summarized in Tables 3 and 4. In Table 3 those values mean that, for example, in the case $N_d = 4$ of Example 1, the obtained neural network can perform exactly the same operations as the target DES does by letting 'true $\geq 1 - 0.01910$' and 'false $\leq -1 + 0.01910$'. In Table 4 those values mean that, for example, in the case $N_d = 4$ of Example 1, the obtained neural network can perform exactly the same operations as the target DES does by letting 'true $\geq 1 - 0.03456$' and 'false $\leq 0 + 0.03456$'.

In order to estimate the effect of the cost function J_2, we perform learning experiments for some different values of ρ and investigate the number of nonzero parameters in the neural networks obtained by the proposed learning method. Table 5 and Table 6 summarize the effect of the cost function J_2 for the case $N_d = 4$ of Example 2, where the number of the parameters which can be treated as zero valued parameters in the obtained RNN and RHONN are shown. In those tables, the maximum absolute values of the tolerances to the threshold values (1 and -1 for RNNs and 1 and 0 for RHONNs) are also shown, and they mean that, for example, in the case $\rho = 0.01$ of Table 5, the obtained neural networks can perform exactly the same operations as the target DES does by letting 'true $\geq 1 - 0.1570$' and 'false $\leq -1 + 0.1570$'. It is observed that the number of zero-valued parameters can be drastically increased by introducing J_2.

6. CONCLUSIONS

In this paper we discussed an identification and realization problem of a class of DESs. We consider a class of DESs which is modeled by finite state automata. In identification and realization of systems by using neural networks, it is essentially important to develop a suitable architecture of neural networks. We presented identification method of the DESs by using two architectures of neural networks which have been proposed by the authors: RNNs and RHONNs. Emphasis was on how to obtain sparse realization, that is, to obtain a neural network with as many zero-valued parameters as possible. It is shown through numerical experiments that the presented method makes it possible to reduce the number of nonzero parameters in the obtain neural networks.

REFERENCES

M. L. Minsky. Computation: Finite and Infinite Machines. *Prentice-Hall*, New York, 1967.

N. Alon, A. K. Dewdney and T. J. Ott, Efficient simulation of automata by neural nets. *Journal of Association for Computing Machinery*, volume 38, No.2, pages 495–514, April 1991.

C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun and Y. C. Lee. Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, volume 4, pages 393–405, 1992.

Table 1. Results of identification of FSA by RNN

	Example 1			Example 2			Example 3		
the number of dynamic neurons	the minimum value of			the minimum value of			the minimum value of		
	J	J_1	J_2	J	J_1	J_2	J	J_1	J_2
$N_d = 1$	0.2036	0.01288	19.07						
$N_d = 2$	0.1971	0.01451	18.26	0.2417	0.01112	23.06	0.6266	0.03517	59.15
$N_d = 3$	0.2169	0.01423	20.26	0.2378	0.01208	22.57	0.6293	0.03682	59.25
$N_d = 4$	0.4296	0.03044	39.91	0.2860	0.01078	27.52	0.8281	0.0400	78.81

Table 2. Results of identification of FSA by RHONN

	Example 1			Example 2			Example 3		
the number of dynamic neurons	the minimum value of			the minimum value of			the minimum value of		
	J	J_1	J_2	J	J_1	J_2	J	J_1	J_2
$N_d = 1$	0.324	0.02367	30.03						
$N_d = 2$	0.3827	0.02668	35.61	0.8075	0.07141	73.61	0.6211	0.04945	57.17
$N_d = 3$	0.4429	0.02757	41.54	0.7719	0.05963	71.23	0.7317	0.04947	68.22
$N_d = 4$	0.4999	0.0289	47.10	0.8101	0.06173	74.84	0.8297	0.05250	77.72

Table 3. The maximum absolute values of the tolerances to the threshold values obtained for RNN

	Example 1	Example 2	Example 3
the number of dynamic neurons	the maximal absolute value of the tolerances	the maximal absolute value of the tolerances	the maximal absolute value of the tolerances
$N_d = 1$	0.01652		
$N_d = 2$	0.01899	0.03851	0.05948
$N_d = 3$	0.02226	0.1316	0.05345
$N_d = 4$	0.01910	0.04568	0.03796

Table 4. The maximum absolute values of the tolerances to the threshold values obtained for RHONN

	Example 1	Example 2	Example 3
the number of dynamic neurons	the maximal absolute value of the tolerances	the maximal absolute value of the tolerances	the maximal absolute value of the tolerances
$N_d = 1$	0.02408		
$N_d = 2$	0.02895	0.1618	0.06824
$N_d = 3$	0.03020	0.1458	0.07415
$N_d = 4$	0.03456	0.1339	0.08534

Z. Zegn, R. M. Goodman and P. Smyth, Learning finite state machines with self-clustering recurrent networks. *Neural Computation*, volume 5, pages 976–990, 1993.

C. L. Giles, D. Chen, G. Sun, H. Chen, Y. Lee and W. Goudreau. Constructive learning of recurrent neural networks: limitations of recurrent casade correlation and a simple solution. *IEEE Trans. on Neural Networks*, volume 6, No.4, pages 829–836, July, 1995.

Y. Kuroe. Representation and identification method of finite state automata by recurrent high-order neural networks. *In* W. Duch et al., editors, *ICANN 2005 Proceedings*, *Lecture Notes in Computer Science*, 3697, pages 181–190, Springer-Verlag, 2005.

Y. Kuroe and Y. Mori. Neural network models for identification and realization of a class of discrete event systems. *In Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, pages 1363–1369, 2007.

E. B. Kosmatopoulos, M. M. Polycarpou, M. A. Christodoulou, and P. A. Ioannos. High-order neural network structures for identification of dynamical systems. *IEEE Trans. Neural Networks*, volume 6, no.2, pages 422–431, Mar. 1995.

Y. Kuroe, H. Ikeda and T. Mori. Identification of nonlinear dynamical systems by recurrent high-order neural networks. *In Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, volume 1, pages 70–75, 1997.

M. Ishikawa. Structual learning with forgetting. *Neural Networks*, volume 9, No.3, pages 509–521, 1996.

Table 5. The number of zero-valued parameters obtained for RNN in Example 2

ρ	the maximal absolute value of the tolerances	the number of zero valued parameters
$\rho = 0$	0.000004519	0
$\rho = 0.00001$	0.001022	150
$\rho = 0.0001$	0.003291	150
$\rho = 0.01$	0.1570	150

Table 6. The number of zero-valued parameters obtained for RHONN in Example 2

ρ	the maximal absolute value of the tolerances	the number of zero valued parameters
$\rho = 0$	0.000004162	0
$\rho = 0.00001$	0.003898	104
$\rho = 0.0001$	0.01285	110
$\rho = 0.01$	0.1382	104